CS206 Data Structure 2014.05.14

Homework #5

Sungwon Cho

1. Representation of the Graph

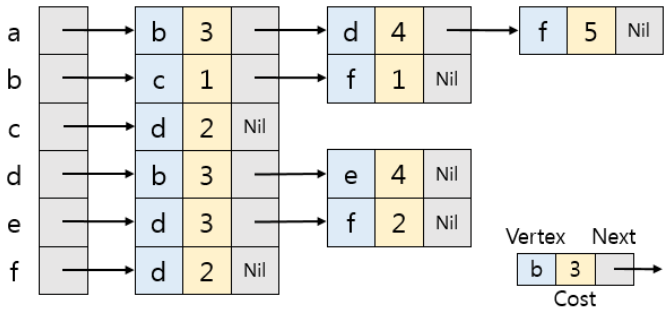
1-(a). Adjacency Matrix

Let A is adjacency matrix of given graph. Then, $A_{ij} = w_{ij}$ where w_{ij} is weight of edge that from vertex i to j. (I use $w_{ij} = \infty$ if such edge does not exist) So, calculating A by this rule get following matrix.

	a	b	c	d	e	f
a	0	3	0	4	0	5
b	0	0	1	0	0	1
c	0	0	0	2	0	0
d	0	3	0	0	4	0
e	0	0	0	3	0	2
f	0	0	0	2	0	0

1-(b). Adjacency List

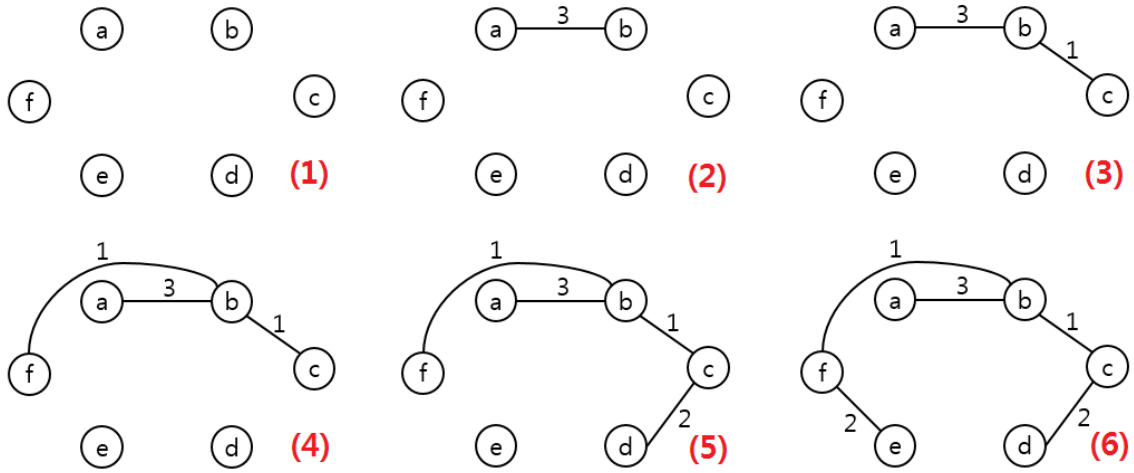
To denote weight of edge, add cost value to each object in linked list. So, each object in linked list have data of vertex, cost and next object where Nil denotes there are no next object. For example, "f 1 Nil" in second row means that there exist edge from **b** to **f** which cost is 1.



2. Minimum Cost Spanning Tree

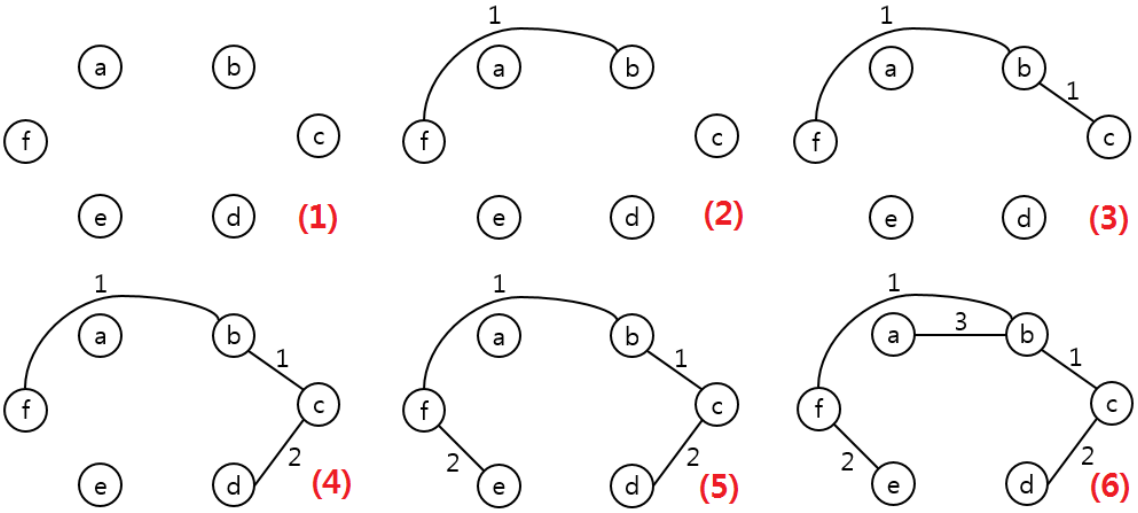
Minimum cost spanning tree does not defined in directed graph, so I converted original graph to undirected graph, and solve this problem on the graph. Also, since in original directed graph, there exist edges $(e, d), (d, e)$, I assume that $w(e, d)$ in undirected graph is 3.

2-(a). Prim's Algorithm



(1) Start with node **a**.
(2) Pick smallest weight edge which connected to **a**, which is (a, b), and add to graph.
↑ $w(a, b) = 3, w(a, d) = 4, w(a, f) = 5$
(3) Repeat choosing step, which is (b, c), and add to graph.
↑ $w(b, c) = 1, w(a, d) = 4, w(a, f) = 5, w(b, f) = 1, w(b, d) = 3$
(4) Repeat choosing step, which is (b, f), and add to graph.
↑ $w(a, d) = 4, w(a, f) = 5, w(b, f) = 1, w(b, d) = 3, w(c, d) = 2$
(5) Repeat choosing step, which is (c, d), and add to graph.
↑ $w(f, d) = 2, w(f, e) = 2, w(a, d) = 4, w(b, d) = 3, w(c, d) = 2$
(6) Repeat choosing step, which is (f, e), and add to graph.
↑ $w(f, e) = 2, w(e, d) = 3$
+ Also, algorithm is terminated since all vertices are in graph.

2-(b). Kruskal's Algorithm



(1) Start with 6 forests.
(2) Pick smallest weight edge, which is (f, b), and add to graph.

(3) Repeat choosing step, which is (b, c), and add to graph.
(4) Repeat choosing step, which is (c, d), and add to graph.
(5) Repeat choosing step, which is (f, e), and add to graph.
(6) Repeat choosing step, which is (a, b), and add to graph.
↑ Here, $(f, d), (e, d), (b, d)$ is ignored since containing those edge make cycle.
+ Also, algorithm is terminated since all vertices are in graph.

3. Transitive Closure

	a	b	c	d	e	f
a	0	1	1	1	1	1
b	0	1	1	1	1	1
c	0	1	1	1	1	1
d	0	1	1	1	1	1
e	0	1	1	1	1	1
f	0	1	1	1	1	1

(0) Define terms: "**u** is reachable from **v**" or "from **v**, **u** is reachable" means that there exist a path from **v** to **u** which length is greater than or equal to 1.
(1) **b**, **c** and **d** are in cycle, so from a vertex, if one of these vertices is reachable, than automatically **b**, **c**, **d**, **e** and **f** are reachable from the vertex.
↑ since there are edges $(b, f), (d, e)$
(2) **a** only has outgoing edges, that means **a** is not reachable from all nodes.
(3) There exist path **a** to **b**, **e** to **d** and **f** to **d**.
↑ since there an edge (a, b), (e, d) and (f, d).
(4) So, by (1), from all vertices there exist a path to **b**, **c**, **d**, **e** and **f**.
(5) Combining (2) and (4) get adjacency matrix representation of the transitive closure.